INF 270
Mandatory Homework I
Due before 10:15 on October 18, 2018

Introduction

In this first mandatory homework we will make a matlab implementation of the simplex method for LP problem on standard form

$$\begin{array}{rr} \max & c^T x \\ \text{such that} & Ax \leq b \\ & x \geq 0. \end{array}$$

where A is an $m \times n$ matrix, c is an n-vector and $b \geq 0$ is an m-vector with nonnegative components. Let $\mathcal{B}$ and $\mathcal{N}$ be the set of indices of basic and nonbasic variables. Note that the vector x is concatenated with the slack variables w such that

$$\mathcal{B} \cup \mathcal{N} = \{1, 2, \ldots, n, n+1, \ldots, n+m\}.$$

Let $A = [A\ I]$ og $c^T = [c^T\ 0]$ where A is expanded with the $m \times m$ identity matrix and c is expanded with m vector consisting of only zeros. In matlab notation `A = [ A eye(m)]` and `c=[c;zeros(m,1)]`. Further, let `bas` be a vector of indices. Then `B = A(:, bas)` is a matrix with as many rows as `A` and with the columns from `A` with indices in `bas`. Similarly, $c_{\mathcal{B}}$ is `c(bas)`.

Please submit for **each sub-exercise (1A, 1B, 1C, 1D, 2A, 2B, 2C, 2D, 3A, 3B, 3C)**
– the corresponding code **on a separate page** and
– your results **on a separate page**.

Exercise 1

As an example we use the LP in Chapter 6 Section **3 An Example**

```
m=3;n=2; c=[4;3];b=[1;3;5]; A=[1 -1; 2 -1; 0 1];
A=[A eye(m)]; c=[c;zeros(m,1)];
bas=n+1:m+n; nbas=1:n;
```

Let `C` be a square nonsingular matrix then `Cinv=inv(C)` is the inverse of `C` (`Cinv=`C^{-1}) and the solution of the linear system `Cy=d` is `y=Cinv*d`.

a) Write a matlab code to compute $x^*_{\mathcal{B}}$ and $x^*_{\mathcal{N}}$ for

A) $\mathcal{B} = \{3, 4, 5\}$

B) $\mathcal{B} = \{1, 4, 5\}$

C) $\mathcal{B} = \{1, 2, 5\}$

D) $\mathcal{B} = \{1, 2, 3\}$

b) Check: Write a matlab code to compute the original variables $x_1, \ldots x_n$ and the slack variables w and check that this is a feasible solution. In addition to $\mathcal{B}$ from 1a) use $\mathcal{B} = \{1, 2, 4\}$. Also compute the value of the objective function `f = c'*x` (in the original variables) for the feasible basic solutions given $\mathcal{B}$. Also compute $c^T_{\mathcal{B}} x^*_{\mathcal{B}}$ and $\zeta^* = c^T_{\mathcal{B}} B^{-1} b$ and compare the results. (The purpose is the check that the indices are used correctly in the matlab program).

Exercise 2

If y is a vector, and we want maximum of y_i this can be computed in matlab using `max(y)`. The index for maximum component is found by `[t,j] = max(y)` and `t = y(j)`$=\max_i y_i$ and `j` is the smallest index where we have the maximal value.

For two vectors x, y with the same number of components maximum of x_i/y_i for all i can be computed from `max(x./y)`. `[t,j] = max(x./y.)` gives `t = x(j)/y(j)`$=\max_i x_i/y_i$ and `j` is the smallest index where we have the maximal value.

To compute $x^*_{\mathcal{B}} = \bar{b}$ we need to compute B as `B = A(:, bas)` and the inverse of B and compute $\bar{b} = B^{-1}b$. To compute $\bar{c} = -z^*_{\mathcal{N}} = c_{\mathcal{N}} - (B^{-1}N)^T c_{\mathcal{B}}$ and $\bar{A} = B^{-1}N$ we form $B^{-1}N$. Note that $B^{-1}N$=`Binv*A(:,nbas)` where `Binv`$=B^{-1}$.

Let `cn = ` $\bar{c}$ (the coefficient in the objective function in the dictionary). In matlab we can make a loop

```
while any cn >0
.....
end
```

which will be executed until all elements in `cn` are **negative** or 0. Use this loop to make a simplex method.

Test the simplex method on

A) Problem 2.1

B) Problem 2.2

C) Problem 2.8

D) the degenerate case from Chapter 3, Section 1 (see page 25).

Exercise 3

We need to expand the simplex method to handle general LP problems on standard form, i.e. for any b.

a) Introduce an auxiliary variable x_0 as in Chapter 2.3. Terminate with a message if the problem is infeasible. Eliminate the auxiliary variable and continue with the simplex method. Note that x_0 may be in basis if the dictionary is degenerate. Test the simplex method on

A) Problem 2.3

B) Problem 2.4

C) Problem 2.7

Please note that the purpose of this exercise is to get experience with the simplex method using the dictionary. This implementation is not efficient and there are many efficient implementations of the simplex method written for matlab available on the net, but please do not use these versions.